Predicting Office Availability Using Crowdsourced Data

Francisco Mateo, David Sobol, Derek Riley and Jesús U. Quevedo-Torrero
Computer Science Department
University of Wisconsin-Parkside
Kenosha WI, 53141
{rileyd, quevedo}@uwp.edu, {mateo001, sobol001}@rangers.uwp.edu

Abstract

Predicting the availability of experts or authorities in their offices is valuable to students, faculty, and other, yet it is difficult due to the inconsistent nature of human behavior and scheduling. Predictive algorithms require many variables such as current schedule, past events, hour of day, habits, and more. The combination of many data sources can still lead to unreliable information, so adding crowdsourcing information is a promising approach. In this work we introduce an application for Android devices that uses crowdsourcing to improve prediction of availability of faculty in their offices. Users of the application can vote on the availability of professors at the current time when they observe it, and then the application predicts the likelihood a professor is available at a specified time by use of Bayesian inference and historical information. Using the wisdom of the crowd we aim to improve interaction of student and professor.

1 Introduction

People in general are often disappointed when their expectations of finding someone during the times they are expected to be available are not met. Whether going to an independent boutique to buy something special, an attorney’s office for advice, or to a professor’s office for help with an assignment; arriving to find no one there is frustrating and discouraging. Having the insight of polling multiple people who have had similar experiences can prevent the wasted time and effort.

Predicting the availability of businesses, professionals, or officials in their offices is not impossible, but it can be costly. Aggregate expenses in terms of hours devoted to monitoring and then the inherent costs of data collection can add up rapidly, especially if multiple parties are involved. Using voluntary input from people who stand to benefit from the information collected would help to have these costs and man-hours circumvented.

In this work, we present a means to improve access to university professors as an illustration for other cases, as mentioned above. This student-created Android application, *ProfTracker*, uses crowdsourcing and historical data to improve the reliability of availability information for a university professor. We apply the wisdom of the crowd combined with historical data using Bayesian inference [1] to give a weighted prediction of the likelihood that a university professor is available during a certain timeframe. ProfTracker uses a simple voting system of Yes/No to collect crowd data as to whether a professor is likely to be in their office at various times. By using historical data from previous semesters, new crowdsourced student data, and applying Bayesian inference, we are able to predict the likelihood that a professor is available at a given time. Bayes’ rule is adapted to utilize cumulative historical data as well as currently gathered data as inputs to produce a probability that a professor will be available. The application ultimately gives students better access to their professor which improves interaction between student and professor.

There are a few difficulties in foreseeing the availability of a professor in his or her office. Privacy issues can be raised when tracking professors’ availability. This problem is alleviated by only including professors on a voluntary basis and professors are made fully aware of the implications of having their hours tracked. Hours tracked are those posted on syllabuses or provided by the professors themselves.

The challenge of motivating crowdsourced workers can be solved through publicizing the benefits of the application. Marketing of the application within the university would improve participation. More participation improves data, which as a result improves the quality of the information that the application provides.

Finally, in this work we also show our preliminary results of using the application. Our results will show that our application can be used to anticipate a professor's availability. We developed an experiment where we tracked a volunteering professor over the course of a few weeks during the summer. We conclude and show that by using data collected from the crowd, we can anticipate the availability of professors can be improved.

The organization of this paper is as follows. In the next section, we present a brief background of crowdsourcing. The following section gives a summary of our modified Bayesian inference formula. Then we describe the case study of our application, and the final section concludes the work.

2 Crowdsourcing

Crowdsourcing is a system for getting information from a swarm of individuals, commonly the general public. The word crowdsourcing is a combination of the two words ‘crowd’ and ‘outsourcing’. The term crowdsourcing is a term that was coined in Wired Magazine by author Jeff Howe in 2006. [11] But crowdsourcing itself has been around since the early 1700s. [11] Crowdsourcing in defined as taking a task and outsourcing it to the crowd or group of people. By breaking up larger more complex tasks we define as ‘macrotasks’ into smaller more manageable ‘microtasks’, and leveraging human ingenuity, crowdsourcing is able to solve complex problems. By crowdsourcing, we are able to help guide computers to develop better solutions than the computers would on their own.

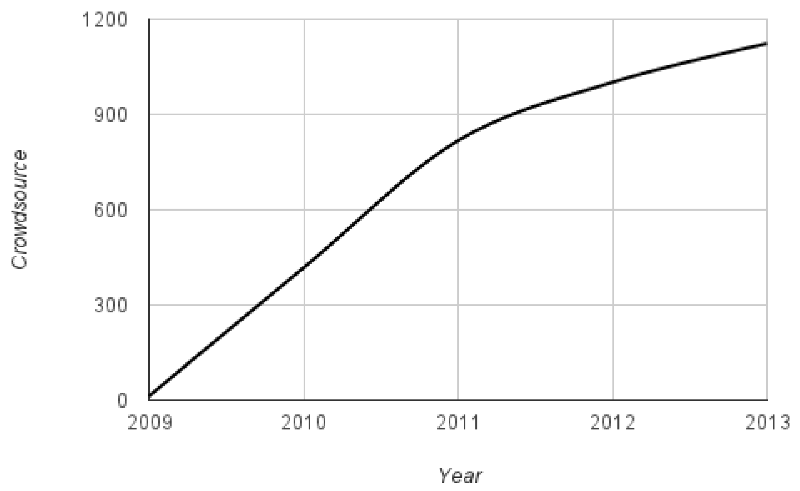


Figure 1: Crowdsource interest over time, from [2].

Interest of crowdsourcing has grown significantly in recent years [2], as shown in Figure 1. The y-axis reflects the total searches that have been done for crowdsourcing. The scale of the y-axis is relative to the total number of searches performed on Google over time. This scale does not represent an absolute value of queries performed on Google because the data collected by Google is normalized and presented on a scale from 0-100. We have calculated the sum of each year and plotted each point on the graph. Businesses in particular have taken an interest in crowdsourcing to develop new products or improve on already existing products. For example Lay’s in 2012 wanted to introduce a new flavor of potato chip, but rather than hiring a marketing firm or similar, Lay’s chose to ask their customers directly for flavor ideas [3].

There are a few challenges developing crowdsourcing approaches. First and foremost privacy issues are almost always a concern. Concerned users can raise issues pertaining to their right of privacy under state and local laws [10]. Another challenge is designing an engaging application that will captivate crowdworkers interests and draw participation can be troublesome. Furthermore, the challenge of assembling the collected data into a meaningful representation which users could understand. Lastly, because crowdsourcing tools often target the general public, they are vulnerable to malicious attacks.

3 The Proftracker Smartphone Application

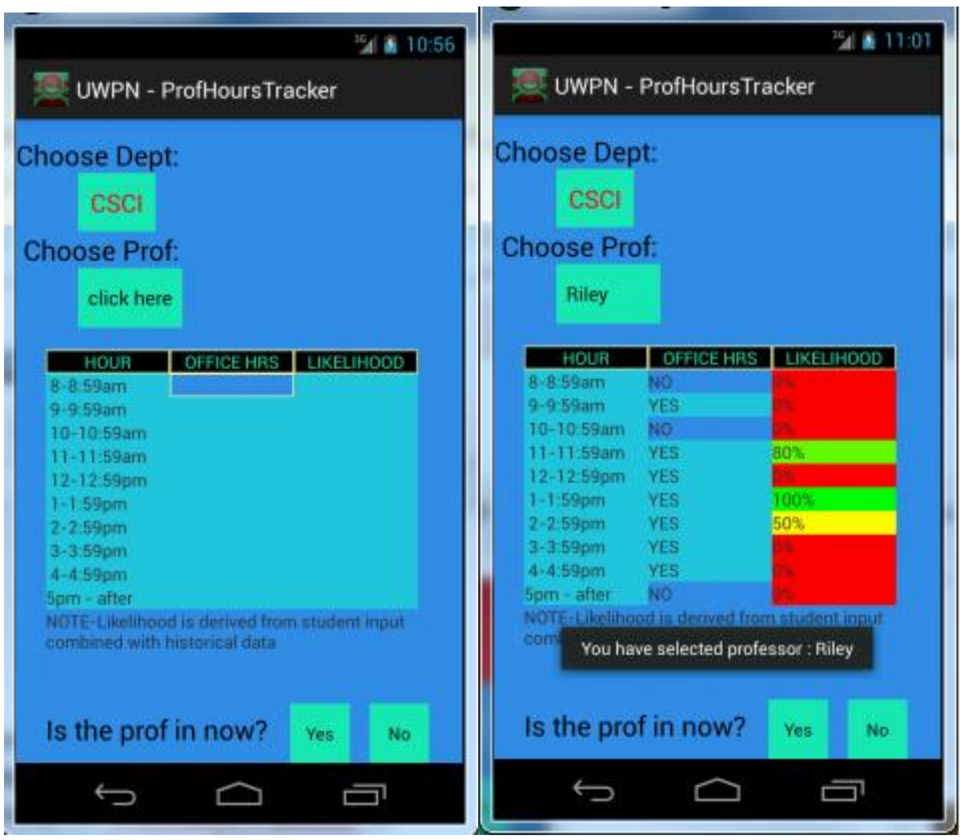
The ProfTracker application is an Android smart device-based prototype comprised of custom Java code and XML code which creates the user interface (UI). ProfTracker relies on a Web server containing a MySQL database that serves as a central repository for storing and serving data. Interaction between the application and the webserver is performed via HTTP requests. The Web server contains a REST (representational state transfer) API built with PHP.

ProfTracker’s main UI is shown in Figure 2. The UI includes (from top) a pull-down menu for department selection, a pull-down menu for professor selection, a table to display a professor’s ‘Office hours’ and ‘Likelihood’ of availability once selected, and a prompt to answer whether that professor is in the office. Operation of the UI is entirely linear; starting from the top a student presses choices and continues to the bottom of the screen.

For example, first, the participant student selects an academic department and then selects a professor from such department. Once a professor has been chosen, a call is made by the application to an external database which returns data that populates 'Office Hours' and ‘Likelihood' in the center-aligned table in the screen. More specifically, the ‘Office Hours’ column contains the value ‘Yes’ or ‘No’ indicating whether a professor has dedicated office time within that hour block. Blocks of one hour each were used for simplicity in this prototype.

On the presumption that the student then goes to visit the professor, they can then answer the prompt located at the bottom of the screen using the 'yes' or 'no' labeled buttons as to whether the professor is in the office at that time. The action of answering the prompt is the crowdsourcing component of the application, since it relies on the collective input of multiple students to derive the probability statistics. Only one vote that a professor is 'in' or 'out' is allowed by the application per clock-hour, per smartphone.

The Office Hours and Likelihood table uses color coding to enhance the values displayed. Split into hours within a range of 8 and 5pm, dedicated office hours are indicated with a lighter shade of blue and are labeled “YES', while undeclared office hours are a darker blue matching the application's background and are labeled 'NO'. Likelihood is rated as a whole-number percentage to enhance readability and also employs color-coding. The range of colors spans from red (0%) to green (100%) and the entire range of values between are displayed as a color appropriate to a place on the spectrum (e.g. 50% is associated with yellow).



HOUR	OFFICE HRS	LIKELIHOOD
8-8:59am		
9-9:59am		
10-10:59am		
11-11:59am		
12-12:59pm		
1-1:59pm		
2-2:59pm		
3-3:59pm		
4-4:59pm		
5pm - after		

NOTE-Likelihood is derived from student input combined with historical data

HOUR	OFFICE HRS	LIKELIHOOD
8-8:59am	NO	0%
9-9:59am	YES	0%
10-10:59am	NO	0%
11-11:59am	YES	80%
12-12:59pm	YES	0%
1-1:59pm	YES	100%
2-2:59pm	YES	50%
3-3:59pm	YES	0%
4-4:59pm	YES	0%
5pm - after	NO	0%

NOTE-Likelihood is derived from student input com

You have selected professor : Riley

Figure 2: ProfTracker Interface: prior to selecting a professor and once selected.

From a structural perspective, data flows between an Android mobile device and the Web server as illustrated in Figure 3. As in the illustration, a user interacts with the application via Android device. We created a custom REST client that performs HTTP requests to a server.

The REST client extends Android AsyncTask which is a must because there cannot be a network operation performed on the main UI thread. Performing a network operation on the main UI thread will cause the app to stall or ‘hang’ as it waits for the network operation to finish. The AsyncTask allows developers proper and easy use of the main UI thread. The AsyncTask class allows developers to perform background operations and return the results on the main UI thread without having to manipulate threads and/or handlers. The REST client also wraps around the java.net package which contains various classes for networked applications. The REST client is responsible for interacting with the REST API on the server. The REST client issues standard HTTP requests to the server. An example of a REST HTTP request is show below in Figure 4. The server houses the REST API which is implemented in PHP. The server responds with a JSON (JavaScript Object Notation) representation of the requested data from the MySQL database. The returned JSON string is then stored as a string variable which is then parsed with our custom JSON client.

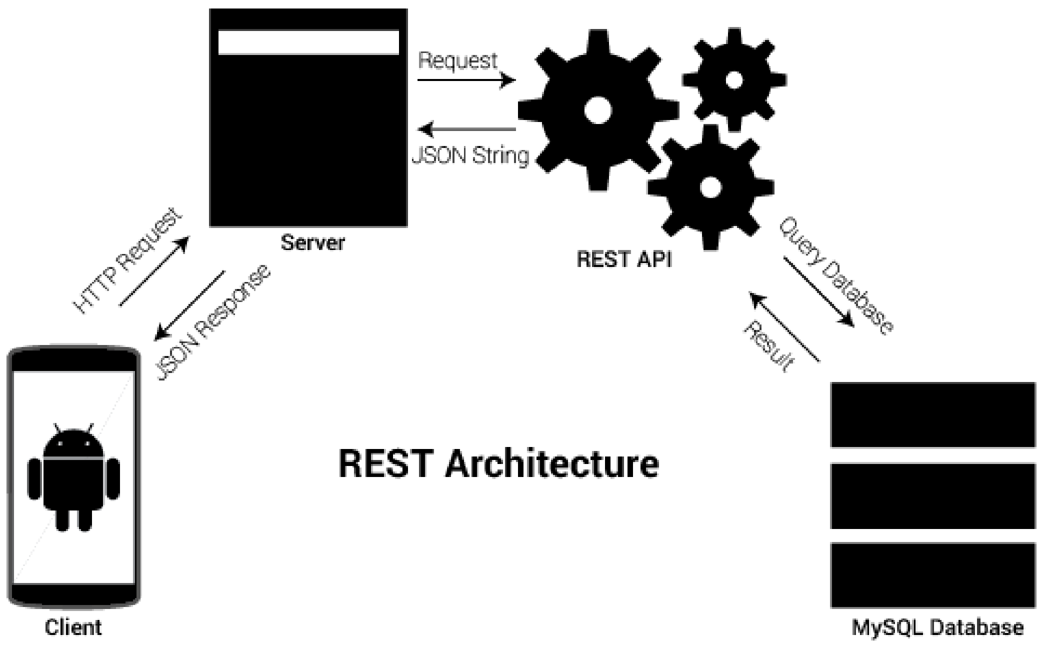


Figure 3: Program Flow Diagram for ProfTracker App.

The JSON client is a small class that wraps around the org.json library that comes standard in the Android SDK. The JSON client takes in a JSON string and parses the JSON string of objects into a standard Java array. The JSON client is instantiated once and used throughout the application.

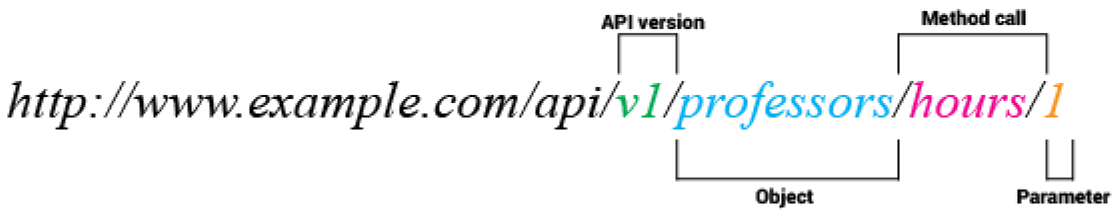
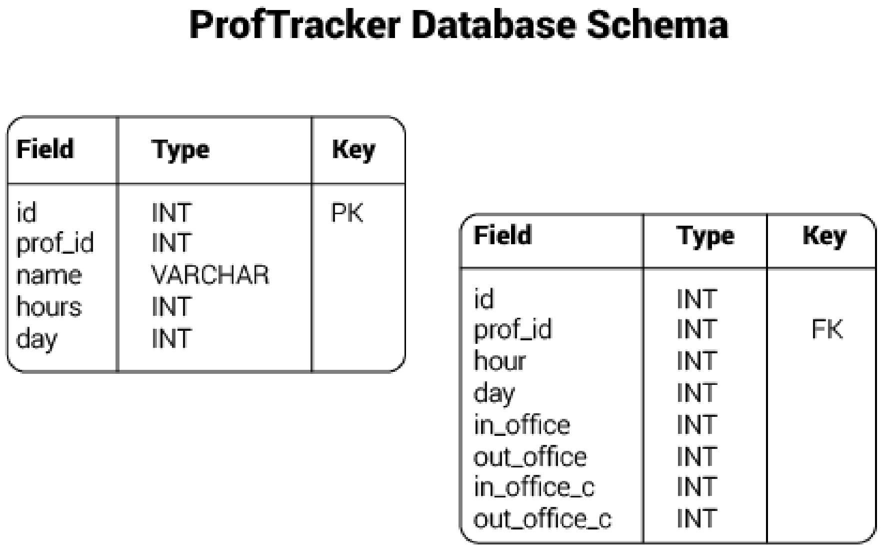


Figure 4: HTTP request to the REST API.

Each time a student responds to the prompt of whether a professor is ‘in office’, a vote is entered into the Notify database. Once new data is inserted into the remote database, the probability of professors' availability is recalculated by a PHP script that resides on the Web server and is triggered by the event. Calculation of the probability will be explained in the next section. Availability probability (as seen under the 'Likelihood' column in the UI) is calculated using a combination of historical data and current data relative to that specific day time frame (see next section).

Professors’ hours and related data are stored within a MySQL database. There are two tables in the Notify database used for our project. The prof_hours and prof_available tables, shown in Figure 5, contain all integer fields. Integer fields were chosen to simplify parsing and storing data.


ProfTracker Database Schema

Field	Type	Key
id	INT	PK
prof_id	INT	
name	VARCHAR	
hours	INT	
day	INT	

Field	Type	Key
id	INT	
prof_id	INT	FK
hour	INT	
day	INT	
in_office	INT	
out_office	INT	
in_office_c	INT	
out_office_c	INT	

Figure 5: Notify Database.

The prof_hours table stores data for a various professors. Specifically, it stores professors office hours, the day of their office hours, and a unique id for them. The prof_available table stores data of the availability of the participating professors, given the foreign key *prof_id*. The prof_available table also stores the hour in which data was inserted into the table as well as the day. The in_office and out_office fields represent the Yes/No voting performed by the crowd. The columns in_office_c and out_office_c are the cumulative totals of the in_office and out_office columns. Together these four integer fields are used in conjunction with our modified Bayesian inference formula to calculate the probability (perct column) a professor is available.

4 Bayesian Predictability Method

Bayesian inference is a method to calculate the probability of an outcome given that prior outcomes have occurred. It is used to quantify a hypothesis and update the probability when new or additional evidence is acquired [1]. Bayesian methods are already being used in a variety of scientific fields such as chemistry [4], biology [5], physics [6], and more. For instance, Bayesian methods have also been used to predict stock returns, reliability of military vehicles, and prediction in other fields as well. [7, 8, 9].

ProfTracker is based on a Bayesian inference model to predict the office availability of a given set of professors. In our model, we use upper case letters to identify sets and lower case letters to identify single elements of a given set. For instance, the participating set of professors is identified by $\boldsymbol{F}$ while $\boldsymbol{f_i}$ identifies a particular professor from $\boldsymbol{F}$, and we refer such professor as Professor $\boldsymbol{i}$. Similarly, Professor $\boldsymbol{m}$ will be addressed as $\boldsymbol{f_m}$. Additionally, this model relies on data collected from student crowd-workers who report the presence or absence of professors that they visit in their offices. Participant students do not need to be enrolled in a particular class to send reports. The set of all crowd-workers is identified by $\boldsymbol{W}$, and $\boldsymbol{W_i}$ refers to the subset of $\boldsymbol{W}$ which are the students enrolled in at least one class taught by Professor $\boldsymbol{i}$ or $\boldsymbol{f_i}$. Then, $\boldsymbol{w}$ represents any student, while $\boldsymbol{w_i}$ represents a student taking a course taught by $\boldsymbol{f_i}$. When student $\boldsymbol{j}$ visits professor $\boldsymbol{i}$'s office, this is recorded as relation $\mathbf{s}_{\boldsymbol{ij}}$ and all these visits are appended to set $\boldsymbol{S_i}$. Hence, set $\boldsymbol{S_i}$ represents all student visits to $\boldsymbol{f_i}$. Each $\mathbf{s}_{\boldsymbol{ij}}$ contains information about the date and time of their visit as well whether or not $\boldsymbol{f_i}$ was present in his office. Time of the visit is mapped into a discrete value $\boldsymbol{H}$ as to the closest hour between 8:00am and 4:00pm, and for any other time value, it will be coded as "5:00pm and after". Date and H are used to uniquely associate a visit report from student $\boldsymbol{j}$ to $\boldsymbol{f_i}$, and therefore, avoid multiple reports during the same time period from the same crowd-worker.

For the purpose of our project, we are interested in calculating the likelihood of a given professor being present in his office when a student visits him for assistance. We call this likelihood "*office availability*", and it will be calculated using the following ratio based on an approximation to the Bayes' formula expression.

$$P_H(f_i|w) = \frac{P_H(w|f_i)P_H(f_i)}{P(w)}$$

$\mathbf{P_H(f_i|w)}$ represents the likelihood of professor $\boldsymbol{i}$ being present in his office given that a student shows there at time H.

$\mathbf{P_H(w|f_i)}$ is the probability that a student visits professor $\boldsymbol{i}$ at time $\boldsymbol{H}$ given that $\boldsymbol{f_i}$ is actually in his office. This probability is estimated as the percentage of students from $\boldsymbol{W_i}$ that are in $\boldsymbol{S_i}$ with visit time recorded as $\boldsymbol{H}$. This also quantifies the likelihood that a student taking a course with $\mathbf{f_i}$ visits him for assistance.

$\mathbf{P_H(f_i)}$ is the probability that professor $\boldsymbol{i}$ is present in his office at time $\boldsymbol{H}$. We calculate this probability as the percentage of crowd-workers in the set $\boldsymbol{S_i}$ that reported affirmatively that professor **i** was at his office during their visit at time $\boldsymbol{H}$.

P(w) is the probability that any student visits professor $\boldsymbol{i}$'s office. We estimate its value as the percentage of students from $\boldsymbol{W}$ that are also in $\boldsymbol{S_i}$ for any value of $\boldsymbol{H}$.

As illustration of our model, let us consider the following scenario, in which, we want to predict the likelihood of Professor Smith being in his office when a student shows there at 2:00pm. Let us assume that the length of $\boldsymbol{W}$ is 40, and the length of $\boldsymbol{W_{Smith}}$ is 25. $\boldsymbol{S_{Smith}}$ contains 12 students' visits from which only 4 out of 6 visits at 2:00pm, report that

Professor Smith was present in his office. Additionally, from those 6 visits only 3 students were enrolled in a class taught by Professor Smith.

Then, $\mathbf{P_{H=2:00pm}(w|f_{Smith})}$ is 3/25, $\mathbf{P_{H=2:00pm}(f_{Smith})}$ is 4/6=2/3 and $\boldsymbol{P(w)}$ is 12/40=3/10

Therefore,

$\mathbf{P_{H=2:00pm}(f_{Smith}|w)= P_{H=2:00pm}(w|f_{Smith})\ \ P_{H=2:00pm}(f_{Smith})} \ / \ \boldsymbol{P(w)= (3/25)(2/3)/(3/10)=20/75}$

5 Experimental Design

To assess the effectiveness of our Android-based app, we recruited two professors from the University of Wisconsin-Parkside to test our prototype. Participating professors submitted voluntarily to monitoring of their availability and were made aware of the purpose of the crowdsourcing project and a description of how data would be used.

Multiple students (the crowd) are used as data sources by means of polling through the app. The students' contributions of whether a professor is in or out of the office is the source data for the calculation of probability using Bayesian inference. A vote of 'Yes' or 'No' from the application is the driving catalyst of the mechanism that results in a calculation for the resulting statistic of 'Availability'.

We developed a Java program that generates random times for a student to visit a professor. The student then launches ProfTracker on their Android device and votes 'Yes' or 'No' based on if that particular professor is in their office. The vote is sent to our database in which the probability is then recalculated with the new vote.

Professors who opted in to the experiment were observed over an agreed-upon range of hours, specifically a three-hour window at four times per hour. During this window of "Office Hours", the professors were recorded as being available or unavailable using the ProfTracker app if they could be visibly seen in or near their respective offices. The action of recording availability was performed by launching the app on a dedicated Android smartphone, selecting the professor from a drop-down menu, and then answering the prompt, "Is the prof in now?" using 'yes' and 'no' buttons within the app interface.

Over a four-day period, we collected data concerning availability of two professors. Using that data as the basis of the historical portion of our probability formula, we predicted the availability of the same two professors within a 40% margin of error. Given the small number of days over which data was collected results were in line, but not as accurate as to what was expected. More specifically, using three day data as a basis, we predicted that one professor would be available 60% of the time relative to published hours. The fourth day, taking data over three hours span, was accurate to within 0%, 20% and 40% error margin respectively. An interesting fringe benefit of the application was that professors availability was more than expected. Outside of posted hours, the professors studied were frequently available in a predictable manner.

6 Conclusions and Future Work

This work shows that by using data collection from crowd-based input, availability of professors within and outside of their published office hours can be predicted to an accuracy of 40% over a small sampling of collected data.

To improve accuracy and add robust features that would make the application more effective and appealing to the user community, we include some ideas for future work. Expanding the timeframe of data collection to multiple semesters could be used to draw conclusions from cumulative data. Accommodating changes in professors' hours midterm is likely to be useful if implementation should occur. Using GPS positioning via the mobile device to determine whether a professor is on campus could improve data. Use of the Google API for calendar data; and alternate views, such as weekly, could be used to see an overall trend or typical pattern that a professor has or will likely exhibit in a future semester. All additional features may be useful for a student who is deciding whether a professor teaching a class that one would take is easy to reach.

Other abstract ideas and values could also be measured similarly, such as polling visitors to a business (e.g. a grocery store) to determine truth in advertising, polling residents of a voting district about their political representative to gauge delivery of campaign promises, etc.

References

[1] Hardas, Manas S., and Lisa Purvis. "Bayesian vote weighting in crowdSourcing systems." Computational Collective Intelligence. Technologies and Applications. Springer Berlin Heidelberg, 2012. 194-203.

[2] "Google Trends - Web Search interest" *Google Trends*. Google, 1 Jan. 2009. Web. 1 Aug. 2014. <http://www.google.com/trends/explore#q=crowdsource>.

[3] Geiger, David, et al. "Crowdsourcing Information Systems-Definition, Typology, and Design." (2012).

[4] Crawford, Sybil L., et al. "Modeling lake-chemistry distributions: Approximate Bayesian methods for estimating a finite-mixture model." *Technometrics* 34.4 (1992): 441-453.

[5] Huelsenbeck, John P., et al. "Bayesian inference of phylogeny and its impact on evolutionary biology." *science* 294.5550 (2001): 2310-2314.

[6] James, Frederick. *Statistical methods in experimental physics*. Vol. 7. No. 4. Singapore: World Scientific, 2006.

[7] Cremers, KJ Martijn. "Stock return predictability: A Bayesian model selection perspective." *Review of Financial Studies* 15.4 (2002): 1223-1249.

[8] Neil, Martin, et al. "Using Bayesian belief networks to predict the reliability of military vehicles." *Computing & Control Engineering Journal* 12.1 (2001): 11-20.

[9] Datta, Gauri Sankar, and Malay Ghosh. "Bayesian prediction in linear models: applications to small area estimation." *The Annals of Statistics* (1991): 1748-1770.
[10] Wang, Yang, Yun Huang, and Claudia Louis. "Respecting User Privacy in Mobile Crowdsourcing." SCIENCE 2.2 (2013): pp-50.
[11] Lynch, Alec. "Crowdsourcing is not new-The History of Crowdsourcing (1714 to 2010)." Retrieved February 26 (2010): 2013.